Simple HTTP Server

In this problem, you need to develop a **web server** that can handle **multiple** incoming HTTP **GET** and **POST** requests and send HTTP response messages back to the client, with appropriate Content-type in the response (according to the standard http protocol). The requirements of the web server are:

- Listen on a specified port, other than the standard port 80, and accept HTTP requests.
- Handle each HTTP request in a separate thread.
- Handle HTTP GET and POST requests. You will be given an http form as a template to demonstrate the POST request.
- Extract filename from HTTP request and return the file or an appropriate error message (e.g. if we type “localhost:8000/index.html” in the address bar of a browser, then we will get the index.html page if found. If not, the browser will show a 404 Not Found Message).
- If the user inputs “localhost:8000” only in the browser, the server must search for an “index.html” file. If found then it will display the file. Otherwise, a 404 Error message must be displayed in the browser as well as in the output.
- Return a HTTP response message and close the client socket connection.
- Return appropriate status code, e.g., 200 (OK) or 404 (NOT FOUND), in the response.
- Determine and send the correct MIME type specifier in the response message.
- **You need to form the response as response_header(status code line, Content-type) + response_body(html file contents).**
- You need to generate an appropriate log file for the corresponding http request.
- **Close the client socket after sending the response.**

You need to implement the basics of http protocol accordingly; therefore, you cannot use any Python high level library(http libraries). You cannot use JavaScript. All the operations should be strictly limited to Socket Programming and html manipulation.

Reference: https://tools.ietf.org/html/rfc7231

Some Guidelines:

- Note that the header lines are separated by newlines. However, newlines in HTTP headers are determined by “\r\n” instead of “\n”.
- You need to include all but only the HTTP Response code line, “Content-Type” and “Content-Length” fields in the response header.
- The “Content-Length” value contains the length of the payload only i.e., excludes the length of the header.
- The header and the body are separated by an extra new line (“\r\n”) after the last header line in the response.
- You need to set the “Content-Type” value properly and for that case you can use libraries of python that determines mime-type from file extension.
- Rendering binary files(images/pdf/video) in response of GET requests is a bonus task for this assignment.
- You should **print request and response messages in the server console** for Debugging.